

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 06-268633

(43)Date of publication of application : 22.09.1994

(51)Int.Cl.

H04L 1/00

(21)Application number : 05-053416

(71)Applicant : FUJITSU LTD

(22)Date of filing : 15.03.1993

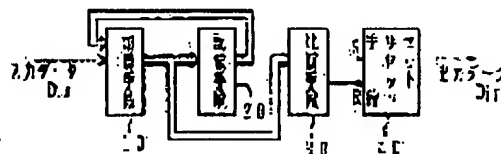
(72)Inventor : OKU TATSUYA

(54) METHOD AND CIRCUIT FOR PROTECTING DATA MULTIPLICITY

(57)Abstract:

PURPOSE: To unnecessitate a large capacity RAM for storing data showing the number of all the objective steps by storing the numbers of continuous errors and non-errors up to the preceding step and adding '1' to the stored value in the case of the next error or non-error input.

CONSTITUTION: As augend values, input data D_{in} and the continuously generated errors/non-errors from a storage means 20 up to the preceding step are inputted to an adding means 10. Corresponding to whether the new input is the error or the non-error, the means adds '1' to the continuous number in both of cases. These continuous numbers are stored in the storage means 20 and at the same time, the added value up to the preceding step on the same time base as these input data is read from the means 20 and inputted to the means 10. Concerning the output of the means 10, set values (n) and (m) at the time of the error and non-error are compared by a comparing means 30. In the case of error continuous number $\geq n$, an error signal is outputted from a set/reset means 40 and in the case of non-error continuous number $\geq m$, an error cancel signal is outputted. The number of errors/non-errors is the number of times of continuous generation and when the continuity is cut even once in the middle, the value of the means 10 is turned to '0'.



LEGAL STATUS

[Date of request for examination]

BEST AVAILABLE COPY

[Date of sending the examiner's decision
of rejection]

[Kind of final disposal of application other
than the examiner's decision of rejection
or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's
decision of rejection]

[Date of requesting appeal against
examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

[Document Name] SPECIFICATION

[Title of the Invention] METHOD AND CIRCUIT OF DATA MULTIPLE PROTECTION

[Scope of Claim for Patent]

[Claim 1] A data multiple protection method for outputting an error signal when bits on the same time axis in former n stages are in an error state and outputting an error reset signal when bits on the same time axis in latter m stages are in a non-error state, wherein,

a number of consecutive errors in bits on the same time axis is calculated for each stage, and an error signal is output when this number reaches n; and a number of consecutive non-errors in bits on the same time axis is calculated for each stage, and an error reset signal is output when this number reaches m.

[Claim 2] A data multiple protection circuit for outputting an error signal when bits on the same time axis in former n stages are in an error state and outputting an error reset signal when bits on the same time axis in latter m stages are in a normal state, comprising:

addition means (10) for, when the bits on the same time axis in the respective stages produce consecutive errors or non-errors, calculating the number of consecutive errors or non-errors by adding one each time such an error or non-error occurs;

storage means (20) for storing an addition value from the addition means (10), and outputting the addition value to the addition means

(10) as a number to be added when the bit on the same time axis in a next stage is input to the addition means (10); comparison means (30) for determining whether the addition value from the addition means (10) is greater than n or m; and set/reset means (40) for, based on a result of the comparison by the comparison means (30), outputting an error signal if the number of consecutive errors is n or greater, and outputting an error reset signal if the number of consecutive non-errors is m or greater.

[Claim 3] The data multiple protection circuit according to claim 2, wherein the addition means (10) includes an addition circuit (10a) for calculating the number of consecutive errors and an addition circuit (10b) for calculating the number of consecutive non-errors.

[Claim 4] The data multiple protection circuit according to claim 2, wherein the comparison means (30) includes a comparator (30a) for determining whether the number of consecutive errors is greater than n or not and a comparator (30b) for determining whether the number of consecutive non-errors is greater than m or not.

[Detailed Description of the Invention]

[0001]

[Technical Field of the Invention] The present invention relates to a method and circuit for data multiple protection.

[0002]

[Prior Art] In the case where even a single error occurring

in data which is received during data transmission results in an error state, the frequency of errors is increased, thus requiring cumbersome maintenance. Moreover, even after a recovery from an error state to a non-error state, it would be dangerous to determine that a normal state has immediately been restored.

[0003] Therefore, a so-called multiple protection technique is adopted, where an error is recognized in the case where an error occurs in bits on the same time axis in the former n stages (n frames), and an error reset is performed unless an error occurs in bits on the same time axis in the latter m stages.

[0004] FIG. 3 shows a conventional circuit which performs the aforementioned multiple protection technique in a case where $n=8$ and $m=4$. A RAM 100 is a multi-terminal RAM, which at least includes $n+m$ input terminals P and output terminals Q , and has a depth corresponding to at least 1 frame (e.g., 1024 bits). To this RAM 100, input data is sequentially written from the terminal P_1 and the terminal P_{10} . The data which has thus been written is sequentially read from the terminal Q_1 (or Q_{10}), beginning from the start bit, and the data having been read is rewritten from the terminal P_2 (or P_{11}). The data which has been output from the terminal Q_2 (or Q_{11}) is input to the terminal P_3 (or P_{12}). Furthermore, the data which has been output from the terminal Q_3 (or Q_{12}) is input to the terminal P_4 (or P_{13}).

[0005] Thus, a single piece of data is sequentially led through the RAM 100, read from each start bit consecutively

for the 8 frames (8 frames of bits on the same time axis are concurrently read out), and a logical sum thereof is derived at a logic AND gate 201. As a result, when an error is occurring in bits on the same time axis for eight consecutive stages, assuming that "1" indicates an error and "0" indicates a non-error, the output from the AND gate 201 is "1", i.e., an error signal, and output from an OR gate 202.

[0006] This output is also input to the input terminal P₉ of the RAM 100, so as to be input from the output terminal Q₉ of the RAM 100 to an AND gate 203. On the other hand, as for the four stages of input data which have been input from the terminals P₁₀ to P₁₃, 4 frames of bits on the same time axis are concurrently read from the output terminals Q₁₀ to Q₁₃, and input to an OR gate 204, which derives a logical sum therebetween. The output from the OR gate 204 is input to the AND gate 203. As a result, if non-errors ("0") continue for four consecutive stages after the aforementioned error signal is output (after the terminal Q₉ has shifted to "1"), the output from the OR gate 204 is "0" and the output from the AND gate 203 is "0". At this time, since the outputs Q₁ to Q₄ of the RAM 100 are "0", as are the outputs Q₁₀ to Q₁₃, the output from the AND gate 201 is also "0" and the output from the OR gate 202 is "0", whereby the error state is reset.

[0007] On the other hand, if the normal state "0" does not continue for four consecutive times, the output from the OR gate 204 is "1" and the output from the AND gate 203 is also "1",

so that the output from the OR gate 202 is "1", whereby the error state is retained.

[0008]

[Problems to be Solved by the Invention] According to the above-described conventional structure, it is necessary to employ a RAM which includes at least $n+m$ input terminals and has a depth corresponding to the number of bits in one frame, thus leading to tremendous cost-wise disadvantages and space-wise disadvantages. Moreover, when the frequency increases, it becomes necessary to perform parallel processing comprising a plurality of such RAMs, so that the cost-wise and space-wise disadvantages will become even more pronounced.

[0009] The present invention is proposed in view of the aforementioned conventional problems, and aims at providing a method and circuit which enables multiple protection of data without employing a large-capacity RAM.

[0004]

[Solution to the Problems of the Invention] In order to attain the above object, the present invention adopts the following means. That is, in a data multiple protection method for outputting an error signal when bits on the same time axis in former n stages are in an error state and outputting an error reset signal when bits on the same time axis in latter m stages are in a non-error state, a number of consecutive errors in bits on the same time axis is calculated for each stage, and an error signal is output

when this number reaches n; and a number of consecutive non-errors in bits on the same time axis is calculated for each stage, and an error reset signal is output when this number reaches m.

[0011] In order to realize the above method, the following circuit is employed according to the present invention, as shown in FIG. 1: addition means (10) for, when bits on the same time axis in the respective stages produce consecutive errors or non-errors, calculating the number of consecutive errors or non-errors by adding one each time such an error or non-error occurs; storage means (20) for storing an addition value from the addition means (10), and outputting the addition value to the addition means (10) as a number to be added when the bit on the same time axis in a next stage is input to the addition means (10); comparison means (30) for determining whether the addition value from the addition means (10) is greater than n or m; and set/reset means (40) for, based on a result of the comparison by the comparison means (30), outputting an error signal if the number of consecutive errors is n or greater, and outputting an error reset signal if the number of consecutive non-errors is m or greater.

[0012] The addition means (10) includes an addition circuit (10a) for calculating the number of consecutive errors and an addition circuit (10b) for calculating the number of consecutive non-errors. The comparison means (30) includes a comparator (30a) for determining whether the number of consecutive errors is greater than n or not and a comparator (30b) for determining

whether the number of consecutive non-errors is greater than m or not.

[0013]

[Function] To the addition means 10, not only input data but also the number of consecutive errors (or non-errors) which have consecutively occurred up to the previous stage of the storage means 20 (described later) is input as a number to be added.

[0014] As a result, if the newly-input data is an error, the addition means 10 increments the number of consecutive errors by one, and if the newly-input data is a non-error, the addition means 10 increments the number of consecutive non-errors by one.

[0015] The number of consecutive errors or non-errors which has been thus obtained is written to and stored in the storage means 20. From the storage means 20, the aforementioned addition value up to the previous stage, being on the same time axis as the data which has been input to the addition means 10, is read and input to the addition means 10.

[0016] The output from the addition means 10 is input to the storage means 20, and also to the comparison means 30. In the comparison means 30, the output is compared against a predetermined setting value, i.e. n in the case of the number of consecutive errors, and m in the case of the number of consecutive non-errors.

[0017] Thus, when the number of consecutive errors $\geq n$ is satisfied, the set/reset means 40 is set so that an error

signal is output. When the number of consecutive non-errors $\geq m$ is satisfied, the set/reset means 40 is reset, thus resulting in an error reset.

[0018] Note that the aforementioned number of errors and number of non-errors are the numbers of consecutively-occurring errors or non-errors. Therefore, even if consecutive errors may have been occurring, a single interrupting non-error will cause the number of errors in the addition means 10 to become zero. Conversely, even if consecutive non-errors may have been occurring, even a single error will cause the number of non-errors in the addition means 10 to become zero.

[0019]

[Embodiment] FIG. 2 is a block diagram showing an embodiment of the present invention. A case is illustrated where an error signal is to be generated when consecutive errors are detected in the former eight stages ($n=8$), and an error reset signal is to be output when consecutive non-errors are detected in the latter four stages ($m=4$).

[0020] Input data D_{in} is input to a terminal B_1 of an adder 10a. On the other hand, from a storage means 20 (described later) to input terminals A_1 , A_2 , and A_3 of the adder 10a, the number of consecutive errors up to the previous round, corresponding to bits on the same time axis as that input to the adder 10a, is input. Therefore, when the input data D_{in} is an error ("1"), the adder 10a outputs a value which is obtained by adding one to the value

from the storage means 20.

[0021] However, it is necessary that the above addition is performed only when errors occur consecutively. Therefore, the output from the adder 10a is further output via AND gates 111, 112, and 113, such that the AND gates 111, 112, and 113 are masked by the input data Din when a non-error ("0") is input. As a result, even if errors have been consecutively input, if a non-error is input before the addition value equals n, the outputs from the AND gates 111, 112, and 113 all become "0", and zero is stored to the storage means 20. Since n=8 is assumed in the present example, the output from the adder 10a only needs to be 3 bits.

[0022] Similarly, the input data Din is inverted via an inverter 9, and the inverted signal (which takes "1" for a non-error) is input to a terminal B₁ of an adder 10b. On the other hand, from the storage means 20 to input terminals A₁ and A₂ of the adder 10b, the number of consecutive non-errors up to the previous round, corresponding to bits on the same time axis as that input to the adder 10b, is input. Therefore, when the input data Din is a non-error, the adder 10b outputs a value which is obtained by adding one to the output from the storage means 20.

[0023] However, it is also necessary in this case that the above addition is performed only when non-errors occur consecutively. Therefore, the output from the adder 10b is further output via AND gates 114 and 115, such that the AND gates 114 and 115 are masked by the output from the inverter 9 when the input

data is an error (the output from the inverter 9 is "0"). Since $m=4$ is assumed in the present example, the output from the adder 10b only needs to be 2 bits.

[0024] The storage means 20 has a depth corresponding to the number of bits in one frame. The number of input and output terminals of the storage means 20 only needs to be at least five, which is a sum of three, i.e., the number of output bits from the adder 10a, and two, i.e., the number of output bits from the adder 10b.

[0025] The output from the adder 10a is written, from the input terminals P_2 , P_3 , and P_4 of the storage means 20, to the respective bit units composing the frame. On the other hand, from the output terminals Q_2 , Q_3 , and Q_4 of the storage means 20, an error addition value up to the previous stage is read from an address corresponding to the bits on the same time axis as that of the data input to the adder 10a, and input to input terminals A_1 , A_2 , and A_3 of the adder 10a as mentioned above. As a result, it becomes possible in the adder 10a to count the number of consecutive errors. The output from the adder 10b is also written, from the input terminals P_5 and P_6 of the storage means 20, to the respective bit units composing the frame. From the output terminals Q_5 and Q_6 of the storage means 20, a non-error addition value up to the previous stage is read from an address corresponding to the bits on the same time axis as that of the data input to the adder 10b, and input to the input terminals A_1 and A_2 of the adder 10b. As

a result, it becomes possible in the adder 10b to count the number of consecutive non-error states.

[0026] The output from the adder 10a is input to a comparator 30a. A setting section 31a sets $n (=8)$ to the comparator 30a. When the input to the adder 10a is greater than eight, a selector 32a is opened and "1" is input to a J terminal of a flip-flop 40_{JK} composing a set/reset means 40.

[0027] On the other hand, the output from the adder 10b is input to a comparator 30b. A setting section 31b sets $m (=4)$ to the comparator 30b. When the input to the adder 10b is greater than four, a selector 32b is opened and "1" is input to a K terminal of the flip-flop 40_{JK}. Thus, the flip-flop 40_{JK}, which was set based on the output from the comparator 30a when eight consecutive errors had occurred, is reset when four consecutive non-errors have occurred.

[0028] The output from the flip-flop 40_{JK} is input to an input terminal P_1 of the storage means 20, so as to be output from an output terminal Q_1 . However, it would also be possible to utilize the output from the flip-flop 40_{JK} itself. Once the bit-by-bit errors or non-errors are thus determined, the circuit in which errors are occurring and the types of errors can be determined, and necessary processes will follow thereafter.

[0029]

[Effects of the Invention] As described above, the present invention provides an effect in that, since the number of

consecutive errors or non-errors up to the previous stage is stored in a storage means, such that one is added to the stored value next time an error or a non-error is input, it is necessary to provide a large-capacity RAM for storing data of all stages of interest. This will also provide greater cost-wise advantages and greater space-wise advantages.

[Brief Description of the Drawings]

[FIG. 1] A block diagram illustrating the principles of the present invention.

[FIG. 2] A block diagram showing one embodiment of the present invention.

[FIG. 3] A block diagram showing a conventional example.

[Description of the Reference Characters]

10(10a,10b) addition means

20 storage means

30(30a,30b) comparison means

40 set/reset means

[Document Name] Abstract

[object] The present invention relates to a method and circuit for data multiple protection, and aims to provide enables multiple protection of data without employing a large-capacity RAM.

[structure] In a data multiple protection method for outputting an error signal when bits on the same time axis in former n stages are in an error state and outputting an error reset signal when bits on the same time axis in latter m stages are in a non-error state, a number of consecutive errors in bits on the same time axis is calculated for each stage, and an error signal is output when this number reaches n ; and a number of consecutive non-errors in bits on the same time axis is calculated for each stage, and an error reset signal is output when this number reaches m .

FIG. 1

DIAGRAM OF PRINCIPLES OF PRESENT INVENTION

INPUT DATA D_{in}
ADDITION MEANS 10
STORAGE MEANS 20
COMPARISON MEANS 30
SET/RESET MEANS 40
OUTPUT DATA D_{in}

FIG. 2

BLOCK DIAGRAM OF PRESENT INVENTION

INPUT DATA D_{in}
OUTPUT DATA D_{in}

10a: ADDER

10b: ADDER

20: STORAGE MEANS

30a: COMPARATOR

30b: COMPARATOR

31b: SETTING SECTION

9: INVERTER

31a: SETTING SECTION

32a, 32b: SELECTORS

40_{JK}: FLIP-FLOP

111-115: AND GATES

FIG. 3

BLOCK DIAGRAM OF CONVENTIONAL EXAMPLE

INPUT DATA Din

OUTPUT DATA Din

100: RAM

201: AND GATE

202: OR GATE

203: AND GATE

204: OR GATE

(19)日本国特許庁(JP)

(12)公開特許公報(A)

(11)特許出願公開番号

特開平6-268633

(43)公開日 平成6年(1994)9月22日

(51)IntCl⁵

H04L 1/00

識別記号

庁内整理番号

F I

技術表示箇所

D 9371-5K

審査請求 未請求 請求項の数 4 O L (全 5 頁)

(21)出願番号 特願平5-53416

(22)出願日 平成5年(1993)3月15日

(71)出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中1015番地

(72)発明者 奥 達也

大阪府大阪市中央区城見二丁目1番61号

富士通関西デジタル・テクノロジー株式会
社内

(74)代理人 弁理士 福井 豊明

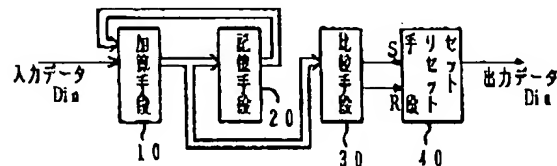
(54)【発明の名称】 データ多重保護方法及び回路

(57)【要約】

【目的】 データ多重保護方法及び回路に関し、大容量のRAMを用いなくてデータの多重保護ができる方法と回路を提供することを目的とするものである。

【構成】 前方n段の同一時間軸上のビットがエラー状態であるとき、エラー信号を出力し、後方m段の同一時間軸上のビットが非エラー状態であるときにエラー解除信号を出力するデータ多重保護方法において、各段の同一時間軸上のビットの連続するエラーの数を算出して、その数がn以上になったときエラー信号を出力し、各段の同一時間軸上のビットの連続する非エラーの数を算出して、その数がm以上になったときエラー解除信号を出力するようにする。

本発明の原理図



【特許請求の範囲】

【請求項1】 前方 n 段の同一時間軸上のビットがエラー状態であるとき、エラー信号を出力し、後方 m 段の同一時間軸上のビットが非エラー状態であるときにエラー解除信号を出力するデータ多重保護方法において、各段の同一時間軸上のビットの連続するエラーの数を算出して、その数が n になったときエラー信号を出力し、各段の同一時間軸上のビットの連続する非エラーの数を算出して、その数が m になったときエラー解除信号を出力するデータ多重保護方法、

【請求項2】 前方 n 段の同一時間軸上のビットがエラー状態であるとき、エラー信号を出力し、後方 m 段の同一時間軸上のビットが正常の状態であるときにエラー解除信号を出力するデータ多重保護回路において、各段の同一時間軸上のビットがエラー又は非エラーを連続して発生するとき、該エラー又は非エラーが発生することに1を加算して、エラー又は非エラーの連続数を算出する加算手段(10)と、

上記加算手段(10)の加算値を記憶するとともに、該加算値を次段の同一時間軸上のビットが上記加算手段(10)に10 入力されるときに被加算値として該加算手段(10)に出力する記憶手段(20)と、

上記加算手段(10)の加算値が上記 n 又は m より大きいかな否かを判断する比較手段(30)と、

上記比較手段(30)による比較の結果、エラーの連続数が n 又はそれより大きいときにエラー信号を出力し、非エラーの連続数が m 又はそれより大きいときにエラー解除信号を出力するセトリセット手段(40)とよりなるデータ多重保護回路。

【請求項3】 上記加算手段(10)がエラーの連続数を算出する加算回路(10a)と、非エラーの連続数を算出する加算回路(10b)とよりなる請求項2に記載のデータ多重保護回路。

【請求項4】 上記比較手段(30)がエラーも連続数が n より大きいかな否かを判断する比較器(30a)と、非エラーの連続数が m より大きいかな否かを判断する比較器(30b)とよりなる請求項2に記載のデータ多重保護回路。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明はデータ多重保護方法及び10 回路に関するものである。

【0002】

【従来技術】 データ伝送において受信したデータに1回でもエラーが出るとエラー状態としてしまうとエラー頻度が高くなり、保守が複雑になる。また、エラー状態から非エラー状態に回復しても直ちに正常な状態になったと判断するのは危険である。

【0003】 そこで、前方 n 段(n フレーム)の同一時間軸にあるビットにエラーが生じたときにエラーとし、後方 m 段の同一時間軸にあるビットにエラーが生じてい15

なければエラーを解除する、いわゆる多重保護方式が採られている。

【0004】 図3は $n=8$ 、 $m=4$ の場合の上記多重保護方式を実施する従来回路を示すものである。RAM100は多端子RAMであって、少なくとも上記 $n+m$ の入力端子P及び出力端子Qを備え、また、少なくとも1フレーム分(例えば1024ビット)の奥行きを備えている。このRAM100に対し端子P₁と端子P₁₀より入力データが順次書き込まれる。このように書き込まれたデータは端子Q₁(又はQ₁₀)より先頭のビットから順に読み出されるとともに、読み出されたデータは端子P₂(又はP₁₁)より再び書き込まれる。また端子Q₂(又はQ₁₁)より出力されたデータは端子P₃(又はP₁₂)に入力され、更に端子Q₃(又はQ₁₂)より出力されたデータは端子P₄(又はP₁₃)に入力される。

【0005】 このように1つのデータは順次RAM100内を巡回し、8フレーム同時に各先頭ビットから順に読み出されて(同一時間軸上の8フレームのビットが同時に読み出されて)、アンドゲート201でその論理和があるビットにエラーが生じているとき、エラーを

“1”、非エラーを“0”とすると、アンドゲート201の出力が“1”、すなわち、エラー信号となってオアゲート202より出力される。

【0006】 この出力はRAM100の入力端子P₄にも入力され、RAM100の出力端子Q₄よりアンドゲート203に入力される。一方、端子P₁₀～P₁₃より入力された4段の入力データは出力端子Q₁₀～Q₁₃より同一時間軸上の4フレームのビットが同時に読み出されて、オアゲート204に入力され、該オアゲート204で論理和がとられる。このオアゲート204の出力は上記アンドゲート203に入力され、これによって、上記エラー信号が出力されてから(端子Q₄が“1”になってから)連続して4段非エラー(“0”)が続くとオアゲート204の出力が“0”となり、アンドゲート203の出力が“0”となる。このときRAM100のQ₁₀～Q₁₃の出力はQ₁₀～Q₁₃の出力と同じ“0”であるので、アンドゲート201の出力も“0”となり、オアゲート202の出力は“0”となってエラー状態は解除される。

【0007】 一方、連続して4回正常な状態“0”が続いていない場合、オアゲート204の出力が“1”となってアンドゲート203の出力も“1”となり、オアゲート202の出力が“1”となってエラー状態を保つことになる。

【0008】

【発明が解決しようとする課題】 上記従来構成によると、少なくとも $n+m$ の入力端子と1フレームのビット数の奥行きのあるRAMを用いる必要があり、コスト上のデメリット、スペース上のデメリットが大きい。ま

た、周波数が大きくなると上記RAMを複数備えた並列処理をする必要があるところから、上記のコスト的、スペース的なデメリットは一層助長されることになる。

【0009】本発明は上記従来の欠点に鑑みて提案されたものであって、大容量のRAMを用いなくてデータの多重保護ができる方法と回路を提供することを目的とするものである。

【0010】

【課題を解決するための手段】本発明は上記目的を達成するために以下の手段を採用している。すなわち、前方n段の同一時間軸上のビットがエラー状態であるとき、エラー信号を出力し、後方m段の同一時間軸上のビットが非エラー状態であるときにエラー解除信号を出力するデータ多重保護方法において、各段の同一時間軸上のビットの連続するエラーの数を算出して、その数がn以上になったときエラー信号を出力し、各段の同一時間軸上のビットの連続する非エラーの数を算出して、その数がm以上になったときエラー解除信号を出力するようにするものである。

【0011】上記方法を実現するために、本発明では以下の回路が用いられる。すなわち、図1に示すように、各段の同一時間軸上のビットがエラー又は非エラーを連続して発生するとき、該エラー又は非エラーが発生することに1を加算して、エラー又は非エラーの連続数を算出する加算手段(10)と、上記加算手段(10)の加算値を記憶するとともに、該加算値を次段の同一時間軸上のビットが上記加算手段(10)に入力されるときに被加算値として該加算手段(10)に出力する記憶手段(20)と、上記加算手段(10)の加算値が上記n又はmより大きいかなかを判断する比較手段(30)と、上記比較手段(30)による比較の結果、エラーの連続数がn又はそれより大きいときにエラー信号を出力し、非エラーの連続数がm又はそれより大きいときにエラー解除信号を出力するセトリセット手段(40)とをそなえる構成とする。

【0012】上記加算手段(10)は、エラーの連続数を算出する加算回路(10a)と、非エラーの連続数を算出する加算回路(10b)とよりなる構成とする。上記比較手段(30)は、エラーの連続数がnより大きいかなかを判断する比較器(30a)と、非エラー連続数がmより大きいかなかを判断する比較器(30b)とよりなる構成とする。

【0013】

【作用】加算手段10には入力データが入力されるとともに、後述する記憶手段20より前段迄に連続して発生したエラー(又は非エラー)の連続数が被加算値として入力されている。

【0014】これによって、新たに入力されたデータがエラーであるときには、加算手段10はエラーの連続数を一つ増加させ、非エラーであるときは非エラーの連続数を一つ増加させる。

【0015】このようにして得られたエラーの連続数又

は非エラーの連続数は記憶手段20に書き込まれて記憶されるとともに、該記憶手段20よりは上記加算手段10に入力されたデータと同一時間軸上の前段迄の上記加算値が読み出されて加算手段10に入力される。

【0016】上記加算手段10の出力は上記記憶手段20に入力されるとともに、比較手段30にも入力され、該比較手段30で所定の設定値、すなわちエラーの連続数の場合はn、非エラーの連続数の場合はmと比較される。

【0017】これによって、エラーの連続数 $\geq n$ になったとき、セトリセット手段40がセットされてエラー信号を出力する。また、非エラーの連続数 $\geq m$ になれば、セトリセット手段40がリセットされてエラー解除となる。

【0018】但し、上記エラーの数、非エラーの数はエラー又は非エラーが連続して発生する回数である。従ってエラーが連続していても途中で1回でも非エラーであると加算手段10のエラー数は零となる。また、逆に非エラーが連続していても途中で1回でもエラーになると、加算手段10の非エラー数は零となる。

【0019】

【実施例】図2は本発明の一実施例を示すブロック図である。ここでは前方8段($n=8$)の連続するエラーを検出したときエラー信号を発生し、後方4段($m=4$)の連続する非エラーを検出したとき、エラー解除信号が出力される場合を示している。

【0020】入力データDinは加算器10aの端子B₁に、入力される。一方、後述する記憶手段20より該加算器10aの入力端子A₁、A₂、A₃に、上記加算器10aに入力されたと同じ時間軸上のビットに対応する前回迄のエラーの連続数が入力されている。従って、入力データDinがエラー("1")であるとき、加算器10aは上記記憶手段20よりの値に1を加えて出力することになる。

【0021】但し、エラーが連続して発生する場合のみ上記加算が行われる必要がある。そこで、上記加算器10aの出力を更にアンドゲート111、112、113を介して出力し、非エラー("0")が入力されたときは入力データDinで上記アンドゲート111、112、113をマスクするようにしている。これによってエラーが連続して入力されていても、加算値がnになる迄に非エラーが入力されたときには、アンドゲート111、112、113の出力は全部"0"となり、記憶手段20には零が記憶されることになる。また、この例では $n=8$ としているので、加算器10aの出力は3ビットで足りる。

【0022】同様にして入力データDinがインバータ9を介して反転され、該反転信号(非エラーのとき"1")が加算器10bの端子B₁に、入力される。一方、記憶手段20より該加算器10bの入力端子A₁、

A_iに上記加算器10bに入力されたと同じ時間軸上のビットに対応する前回迄の非エラーの連続数が入力されている。従って、入力データD_{in}が非エラーであるとき、加算器10bより上記記憶手段20よりの出力に対して1が加えられて出力される。

【0023】但し、この場合も非エラーが連続して発生した場合のみ上記加算を行う必要があるので、上記加算器10bの出力を更にアンドゲート114、115を介して出力し、該アンドゲート114、115は入力データがエラー（インバータ9の出力が“0”）のとき、該インバータ9の出力でマスクされるようになっている。また、この例ではm=4としているので、この加算器10bの出力は2ビットで足りる。

【0024】記憶手段20は実行きが1フレームのビット数に対応し、入力及び出力の端子数はこの例の場合には、少なくとも上記加算器10aの出力ビット数3と、加算器10bの出力ビット数2の合計5あれば足りる。

【0025】上記加算器10aの出力は上記記憶手段20の入力端子P₁、P₂、P₃よりフレームを構成する各ビット単位に書き込まれる。一方、この記憶手段20の出力端子Q₁、Q₂、Q₃よりは加算器10aに入力されたデータと同一時間軸上のビットに対応するアドレスより、前段迄のエラー加算値が読み出されて、上記のように加算器10aの入力端子A₁、A₂、A₃に入力される。これによって加算器10aではエラーの連続数の計数が可能となる。加算器10bの出力も記憶手段20の入力端子P₁、P₂よりフレームを構成する各ビット単位に書き込まれる。また、記憶手段20の出力端子Q₁、Q₂よりは上記加算器10bに入力されているデータと同一時間軸上のビットに対応するアドレスより、前段迄の非エラー加算値が読み出され、加算器10bの入力端子A₁、A₂に入力される。これによって、加算器10bでは非エラー状態の連続数の計数が可能となる。

【0026】上記加算器10aの出力は比較器30aに入力される。この比較器30aには設定部31aよりn(=8)が設定され、加算器10aの入力が8より大き

いとき、セクタ32aを開いてセトリセット手段40を構成するフリップフロップ40₁のJ端子に“1”を入力する。

【0027】一方、上記加算器10bの出力は比較器30bに入力される。この比較器30bには設定部31bよりm(=4)が設定され、加算器10bの入力が4より大きいとき、セクタ32bを開いてフリップフロップ40₂のK端子に“1”を入力する。これによって、エラーが8回連続して発生した時、上記比較器30aの出力でセットされたフリップフロップ40₁が、非エラーが4回連続して発生することによって、リセットされることになる。

【0028】このフリップフロップ40₁の出力は上記記憶手段20の入力端子P₁に入力され、出力端子Q₁より出力される。ただし、上記フリップフロップ40₁の出力をそのまま利用することも可能である。このようにビット毎のエラー又は非エラーが判明すると、エラーの生じている回線やエラーの種類が判ることになり、この後は必要な処理がとられる。

【0029】

【発明の効果】以上説明したようにこの発明は、前段迄のエラー又は非エラーの連続数を記憶手段に記憶して、次にエラー又は非エラーが入力されたときに上記記憶値に1を加算するようにしているので、対象となる全段数のデータを記憶するための大容量のRAMを必要としない効果がある。従って、価格メリット及びスペースメリットも大きくなる効果がある。

【図面の簡単な説明】

【図1】本発明の原理ブロック図である。

【図2】本発明の一実施例ブロック図である。

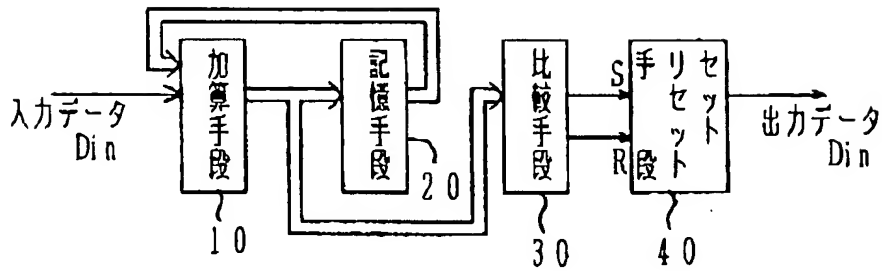
【図3】従来例ブロック図である。

【符号の説明】

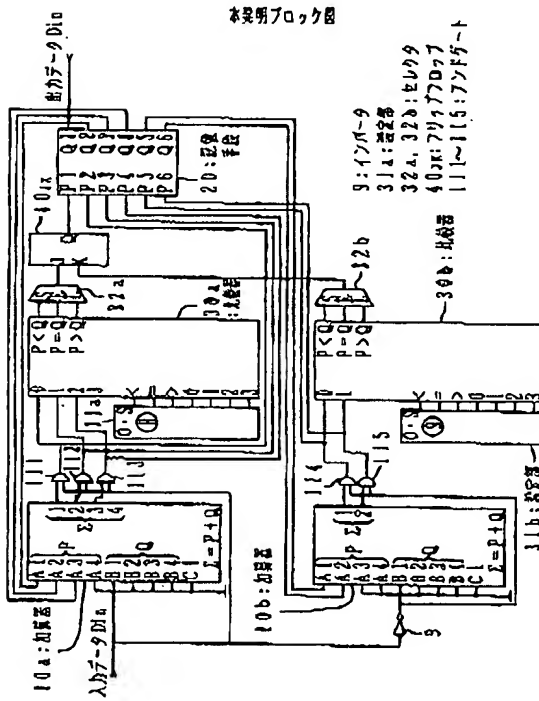
10 (10a, 10b)	加算手段
20	記憶手段
30 (30a, 30b)	比較手段
40	セトリセット手段

【図1】

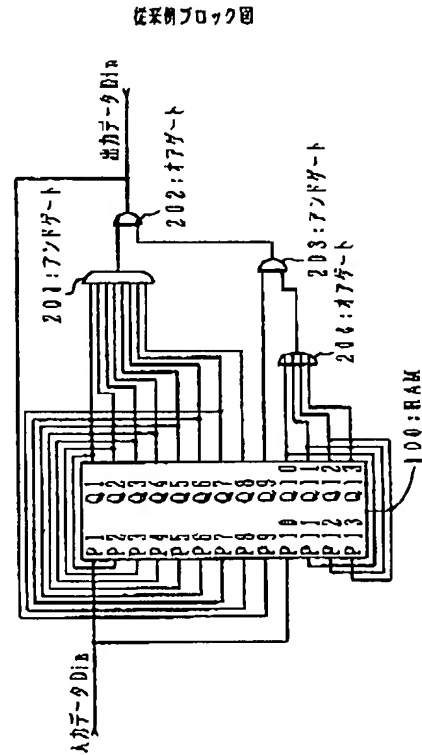
本発明の原理図



【図2】



【図3】



**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.